REMARKS

In the Office Action, Claims 1-36 are pending and presented for examination. Claims 1-4, 8-14, 18-21, 26 and 31-36 are rejected and Claims 5-7, 15-17, 22-25 and 27-30 are objected to. In this Response, Claims are amended, no claims are cancelled, and no claims are added. Applicants respectfully request reconsideration of pending Claims 1-36 in view of the following remarks.

I. Claim Rejections Under 35 U.S.C. §103

Claims 1, 11, 21, 26, 31 and 34 are rejected under 35 U.S.C. §103(a) as being unpatentable over Lakshmanamurthy et al. *"Network Processor Performance Analysis Methodology"*, Aug. 15, 2002, Intel Technology Journal ("Lakshmanamurthy") in view of U.S. Patent 7,392,511 to Brokenshire et al., ("Brokenshire"). Applicants respectfully traverse this rejection.

Claim 1 recites:

> 1. A method comprising:
> configuring one or more processors into a D-stage processor pipeline;
> transforming a sequential network application program into D-pipeline stages that collectively perform an infinite packet processing stage (PPS) loop of the sequential network application program; and
> executing the D-pipeline stages in parallel within the D-stage processor pipeline to provide parallel execution of the infinite PPS loop of the sequential network application program. (Emphasis added.)

While Applicant's argument here is directed to the cited combination of references, it is necessary to first consider their individual teachings, in order to ascertain what combination (if any) could be made from them.

Lakshmanamurthy relates to a network processor performance analysis methodology for analyzing the performance of networking applications targeted for the IXP 2400 network processor. (See Abstract.) In contrast with Claim 1, Lakshmanamurthy does not disclose or suggest transforming a sequential network application program into D-pipeline stages that collectively perform an infinite packet processing stage (PPS) loop of the sequential network

application program, as in Claim 1. Lakshmanamurthy does disclose a data movement model for estimating compute cycles and total I/O references required for the various operations performed by a network processor on each received packet; Lakshmanamurthy further disclosed the estimation of the total budget for the packet processing to determine how functional blocks are mapped onto available hardware resources, and how software concepts are used to meet the performance goals, where the methodology is validated by implementing microcode and tuning the code (on a simulator and hardware) to demonstrate line-rate performance. (See page 20, left column, lines 10-47.)

Apposite to Claim 1, the disclosure of Lakshmanamurthy is directed to providing a performance analysis of a hypothetical target application if implemented to use the parallel architecture of an IXP 2400 network processor. Lakshmanamurthy does disclose a line-rate performance based on a data movement model, an estimated number of compute cycles, and total I/O references required for operations performed on a packet basis, and a total available budget for packet processing, to enable performance analysis of a hypothetical target application that can be written to run on an IXP 2400 network processor (see *supra*). Nevertheless, the disclosure of Lakshmanamurthy is something completely different from transforming a sequential network application program into D-pipeline stages that collectively perform a sequential packet processing stage (PPS) of the sequential network application, as in Claim 1.

As correctly recognized by the Examiner, Lakshmanamurthy fails to disclose or suggest transforming a sequential network application program into D-pipeline stages that collectively perform an infinite packet processing stage (PPS) loop of the sequential network application and executing the D-pipeline stages in parallel within the D-stage processor pipeline to provide parallel execution of the infinite PPS loop of the sequential network application program, as in Claim 1. As a result, the Examiner cites Brokenshire. However, the Examiner's citing of Brokenshire fails to rectify the above deficiencies of Lakshmanamurthy.

Brokenshire relates to a computer system where members of a network contain computing modules that have a processing controller and a plurality of identical processing units, for performing parallel processing of the data and applications transmitted over the network (see col. 4, line 58 to col. 5, line 3). Brokenshire refers to this configuration as a homogeneous

configuration, where the particular computing device performing the actual processing of data and applications is unimportant. To facilitate such homogeneous configuration, Brokenshire describes software cells to take advantage of processing speeds and efficiencies facilitated by the system. As described by Brokenshire, the data and applications processed by the system are packaged into uniquely identified, uniformly formatted software cells (see col. 5, lines 4-26). As further described by Brokenshire, the uniform structure and unique identification of software cells facilitates processing of applications and data on any computer or computing device of the network; specifically, software cells can migrate throughout the network for processing on the basis of availability of processing resources on the network (see col. 5, lines 26-38).

In contrast with Claim 1, Brokenshire does not disclose or suggest transforming a sequential network application program into D-pipeline stages that collectively perform an infinite PPS loop of a sequential network application, as in Claim 1. In Brokenshire, the software cells are not required to execute in parallel, nor do they collectively perform an infinite PPS loop of a sequential network application. As a result, the execution of the software cells of Brokenshire cannot properly be interpreted as teaching transforming a sequential network application program into D-pipeline stages that collectively perform an infinite PPS loop of the sequential network application program, as in Claim 1.

In contrast with the infinite PPS loop of a sequential network application program, as in Claim 1, the software cell processing described by Brokenshire performs multiple software cell processing independently, and not collectively, as in Claim 1. Brokenshire teaches that the limited processing power of clients requires transmission of independent software cells to migrate through a network for processing by an available resource (see col. 5, lines 33-38). As a result, Brokenshire cannot rectify the failure of Lakshmanamurthy to disclose or suggest transforming a sequential network application program into D-pipeline stages that collectively perform a sequential packet processing stage (PPS) of the sequential network application program, as in Claim 1.

According to the Examiner, transforming a sequential network application program into D-pipeline stages that collectively perform an infinite PPS loop in the sequential network application program is disclosed by Brokenshire at col. 6, lines 15-28 and col. 17, lines 5-23 and

col. 19, lines 43-54 with reference to Figs. 39, 40, and 40B. We respectfully disagree with the Examiner's assertions and characterizations regarding Brokenshire.

The passages referred to by the Examiner describe processing units (PU) that are the basic processing module for all members of the network described by Brokenshire (see col. 5, lines 47-48). As described by Brokenshire, the PU can be a standard processor, capable of standalone processing, that is used to schedule and orchestrate the processing of data and applications by synergistic processing units (SPUs). As indicated by Brokenshire, the SPUs are preferably single-instruction multiple-data (SIMD) processors, which, under the control of a PU, perform processing of data and applications in a parallel and independent manner (col. 6, lines 15-21).

Although the passages referred to by the Examiner describe functionality performed by SPUs under the direction of a PU, the SPUs described by Brokenshire do not collectively perform an infinite packet processing stage (PPS) loop of a sequential network application program. Brokenshire does disclose a network spulet that comprises a program for processing a network protocol of a network which is used for packetizing as well as de-packetizing software cells (see col. 17, lines 47-55). We submit that it is improper for the Examiner to rely on Brokenshire to disclose collectively performing an infinite PPS loop of a sequential network application program, since the network spulet as described by Brokenshire is used solely to enable software cells to migrate throughout the network for processing on the basis of the availability of processing resources of the network (see col. 5, lines 34-38).

Hence, neither the passages referred to by the Examiner, nor any other portions of Brokenshire, can disclose or suggest transformation of a sequential network application program into D-pipeline stages that collectively perform an infinite PPS loop of the sequential network application program, as in Claim 1. Furthermore, the migration of software cells throughout the network described by Brokenshire, for processing on the basis of the availability of processing resources on the network, does not disclose or suggest executing the D-pipeline stages in parallel within the D-stage processor pipeline to provide parallel execution of the infinite PPS loop, as in Claim 1.

According to the Examiner, execution of the D-pipeline stages is disclosed by col. 2, lines 36-52, col. 10, lines 10-22, and col. 18, lines 23-4 with references to FIGS. 1, 39, and 40B. However, these passages merely refer to the techniques that are used to detect packetized software cells that migrate throughout the network for processing on the basis of the availability of processing resources on the network. The de-packetizing of transmitted software cells using the TCP/IP network protocol cannot be properly interpreted as execution of D-pipeline stages in parallel within a D-stage processor pipeline to provide parallel execution of an infinite PPS loop of a sequential network application program, as in Claim 1. In fact, de-packetizing of software cells discloses neither an infinite PPS loop of a sequential network application program, nor the parallel execution of such PPS loop of a sequential network application program, as in Claim 1.

Hence, no combination of Lakshmanamurthy and Brokenshire could disclose, teach, or suggest transforming a sequential network application program into D-pipeline stages that collectively perform an infinite packet processing stage (PPS) loop of the sequential network application, much less parallel execution of the infinite PPS loop of the sequential network application program, as in Claim 1.

For each of the above reasons, therefore, Claim 1 and all claims which depend from Claim 1 are patentable over the cited art.

Each of Applicant's other independent claims, including Claims 11, 21, 26, 31, and 34, recite features similar to those highlighted above in Claim 1. Therefore, all of Applicant's other independent claims, including Claims 11, 21, 26, 31, and 34, and all claims which depend from them, are also patentable over the cited art for similar reasons. Consequently, please reconsider and withdraw the §103(a) rejection of Claims 1, 11, 21, 26, 31, and 34.

Claims 2, 12, 32, and 35 are rejected under 35 U.S.C. §103(a) as being unpatentable over Lakshmanamurthy in view of Brokenshire and further in view of Rakhmatov et al. *"Hardware-Software Bipartitioning for Dynamically Reconfigurable Systems", May 2002, ACM* ("Rakhmatov"). **Claims 3-4, 8, 10, 13-14, 18, 20, 33, and 36** are rejected under 35 U.S.C. §103(a) as being unpatentable over Lakshmanamurthy in view of Brokenshire and Rakhmatov and further in view of Robschink et al. (*"Efficient Path Conditions in Dependence Graphs", May 2002, ACM*) ("Robschink"). **Claims 9 and 19** are rejected under 35 U.S.C. §103(a) as being

unpatentable over Lakshmanamurthy in view of Brokenshire, Rakhmatov, and Robschink and further in view of Goldberg et al. "*A New Approach to the Maximum-Flow Problem", 1988, ACM* ("Goldberg"). Applicants respectfully traverse these rejections.

DEPENDENT CLAIMS

In view of the above remarks, a specific discussion of the dependent claims is considered to be unnecessary. Therefore, Applicant's silence regarding any dependent claim is not to be interpreted as agreement with, or acquiescence to, the rejection of such claim or as waiving any argument regarding that claim. Consequently, please reconsider and withdraw the §103(a) rejection of dependent Claims 2-4, 8, 10, 12-14, 18, 20, 32, 33, and 35.

II. Allowable Subject Matter

Claims 5-7, 15-17, 22-25, and 27-30 are objected to as being dependent upon a rejected base claim, but would be allowable if rewritten to overcome the rejections under 35 U.S.C. §103(a) set forth to include all the limitations of the base claim and any intervening claims.

Applicants respectfully thank the Examiner for recognizing the allowability of Claims 5-7, 15-17, 22-25, and 27-30. However, for at least the reasons provided above, Applicants respectfully submit that such claims, based on their dependency from independent Claims 1 and 21, are also patentable over Lakshmanamurthy, Brokenshire, Rakhmatov, and Robschink and further in view of Goldberg as well as the references of record. Therefore, Applicants respectfully request that the Examiner reconsider and withdraw the objection to Claims 5-7, 15-17, 22-25, and 27-30, and allow such claims, based on their dependencies from Claims 1 and 21.

CONCLUSION

In view of the foregoing, it is believed that all claims now pending (1) are in proper form, (2) are neither obvious nor anticipated by the relied upon art of record, and (3) are in condition for allowance. A Notice of Allowance is earnestly solicited at the earliest possible date. If the Examiner believes that a telephone conference would be useful in moving the application forward to allowance, the Examiner is encouraged to contact the undersigned at (310) 207-3800.

If necessary, the Commissioner is hereby authorized in this, concurrent and future replies, to charge payment or credit any overpayment to Deposit Account No. 02-2666 for any additional fees required under 37 C.F.R. §§ 1.16 or 1.17, particularly extension of time fees.

Respectfully submitted,

BLAKELY, SOKOLOFF, TAYLOR & ZAFMAN LLP

Date: March 25, 2009

Joseph Lutz, Reg. No. 43,765

1279 Oakmead Parkway
Sunnyvale, California 94085-4040
Telephone (310) 207-3800
Facsimile (408) 720-8383

CERTIFICATE OF TRANSMISSION
I hereby certify that this correspondence is being submitted electronically via EFS Web on the date shown below to the United States Patent and Trademark Office.

Alexandra Y. Caluen — March 30, 2009